CONFIGURABLE FAST ETHERNET AND GIGABIT ETHERNET DATA PORT

Related applications

The present rule is a group of five patent applications having the same priority date. The present application relates to an switch having an ingress port which is configurable to act either as eight FE (fast Ethernet) ports or as a GE (gigabit Ethernet port). Application PCT/SG02/----- relates to a parser suitable for use in such as switch. Application PCT/SG02/----- relates to a flow engine suitable for using the output of the parser to make a comparison with rules. Application PCT/SG02/----- relates to monitoring bandwidth consumption using the results of a comparison of rules with packets. Application PCT/SG02/----- relates to a combination of switches arranged as a stack. The respective subjects of the each of the group of applications have applications other than in combination with the technology described in the other four applications, but the disclosure of the other applications of the group is incorporated by reference.

Field of the invention

The present invention relates to a data port, such as an ingress/egress data port of an Ethernet switch, which is configurable to operate in one of multiple modes.

Background of Invention

A gradual shift is occurring from Fast Ethernet (FE) systems to Gigabit Ethernet (GE) systems. This, and the use of Ethernet in First Mile (EFM), makes it advantageous to provide a configurable Ethernet switch which can function both as a Fast Ethernet and as a Gigabit Ethernet switch, in order to facilitate the transition from FE to GE Ethernet.

Summary of the Invention

In general terms, the present invention proposes that an ingress/egress port is operable in two modes, in a first mode as a GE port and in a second mode as a plurality of FE ports. The port includes a plurality of MAC interfaces each of which is capable of receiving/transmitting FE packets. At least one of the MAC interfaces is further configurable to receive GE packets. The port further includes receive and transmit modules which are configurable respectively to receive both GE and FE packets from, and transmit GE and FE packets to, the interfaces.

Preferably, there is only a single MAC interface which is configurable to receive/transmit either GE or FE packets, and the other MAC interfaces are only adapted to receive/transmit FE packets. Typically, there are 8 MAC interfaces per port.

Preferably, the MAC interfaces are each associated with a buffer, so that, as packets are received by the interfaces they are stored. Packets may arrive concurrently on all FE ports and are stored in individual buffers. The buffers are polled sequentially by the receive module.

Preferably, there are a plurality of such ports in the Ethernet switch. For example, if there are 8 such ports, then by switching different numbers of the ports between the two modes, the switch may operate in 9 different modes: as

8 GE ports, 7 GE ports and 8 FE ports, ..., 2 GE ports and 48 FE ports, 1 GE port and 56 FE ports, or simply as 64 FE ports.

Brief Description of The Figures

Preferred features of the invention will now be described, for the sake of illustration only, with reference to the following figures in which:

Fig. 1 shows schematically a configurable Ethernet port which is an embodiment of the invention;

Fig. 2 shows the architecture of the receive module of Fig. 1;

Fig. 3 shows the concept of storage within the receive module of Fig. 2;

Fig. 4 shows a timechart showing how the receive module of Fig. 2 processes a packet; and

Fig. 5 shows the architecture of the transmit module of Fig. 1.

Detailed Description of the embodiments

Referring firstly to Fig. 1, a configurable Ethernet port 1 according to the invention comprises an interface 3 having the appropriate number of pins to function as 8 FE MAC interfaces. These are pins are partitioned into 8 sets of pins. One of sets of pins can function as an interface for GE packets, since such a packets requires the same number of pins as an FE packet. The interface 3 receives/transmits GE packets (e.g. in the GMII, RGMII or TBI formats) from/to the left of the figure. The interface 3 transmits received packets to a receive module 5, and transmits to the left of the figure packets received from a transmit module 7.

The receive module 5 is shown including the following major sections: a memory 51 rx_mem; a 53 section having an interface rx_max_iface for extracting the Ethernet header from incoming data packets and putting it into a descriptor, and for putting data from incoming packets into cells, and a

parser rx_parser for extracting information from other headers in the packet, and adding this information also to the descriptor; and a section 55 having a first interface ax_ed_iface and a second interface rx_pr_iface

The Ethernet port 1 further includes a transmit module 7, shown having the following major sections: a memory TX_MEM 71; a TX_MAC_IFACE 73; and a section 75 including a tx_edram_iface and a tx_pq_iface.

The receive module 5 communicates with a packet resolution module 2 and an embedded DRAM (eDRAM) 4. The transmit module 7 communicates with the embedded DRAM 4 and a queue/packet manager 6.

The Ethernet port 1 further includes an interface 9 to the CPU, and MIB (management information base) counters 11. These counters count various packet types, e.g. the number of 64 byte packets.

The structure of the RX module 5 is shown in more detail in Fig. 2. The basic data/control flow of the RX module is as follows:

1. A set of first-in-first-out input buffers rx_ifofo receives and stores incoming packets and handles the interface to the MACS 3.

2. rx_ifofo informs the receive MAC interface rx_mac_iface when data is ready.

3. The interface rx_mac_iface fetches data from rx_ififo and stores it in the memory rx_mem_mgr. For each packet fetched by rx_max_iface, a respective packet descriptor (PKT_DSC) is allocated to the packet. The packet descriptor contains status information about the incoming packet (control information received from IFIFO/Current packet number etc). This packet descriptor is allocated to an incoming packet and only de-allocated when the entire packet has been written to the memory. Packet desriptors are located within the register file rx_reg_file. Rx_reg.file and rx_mem_mgr are part of

rx_mem in Fig. 1. In all 24 packet descriptors are available. Thus, in FE mode, 3 packet descriptors are available per FE port. The packet descriptor is shown in the top line of Fig. 3, as PKT_DSC/PH/RKEY(#0) etc.

4. The first 24 bytes of each frame are stored within a separate memory bank PKT_PR_DATA also within RX_MEM to be sent to the packet resolution module 2. This is illustrated in Fig. 3, which shows that there are sections of this memory (numbered PKT_PR_DAT(#0) to PKT_PR_DAT(#23)), i.e. one corresponding to each packet descriptor. These 24 bytes are the last to be sent to the eDRAM 4 along with the packet header (PH). After the first 24 bytes, each 32byte chunk is stored in a separate 32 byte memory cell (RXD_CELL). These cells are allocated to the incoming data in a round-robin fashion. Cells are freed when the corresponding data has been written to the eDRAM. In all 32 memory cells RXD_CELL are available, i.e. 4 per FE port. In Fig. 3 allocation is shown for a 104 byte frame, which is placed into the region of Fig. 3 which is shown shaded. As mentioned above, the first 24 bytes are stored in the memory section PKT_PR_DATA#0 while the remaining 80 bytes is stored in 3 memory cells (RXD_CELL#0, RXD_CELL#1 and RXD_CELL#02), leaving the third memory cell RXD_CELL#2.

5. Incoming data is analysed by the RX parser rx_parser, which extracts eight 2-byte items, called IKEY, per packet. These 2-byte items are written to the first 16 bytes of PKT_PR_DATA. The extraction is performed based on the offsets programmed in the 8 Offset registers of a register file. The rx-parset also determines whether a frame is VLAN tagged and/or SNAPped. The VLAN tag and priority are also extracted if present. This information is sent to the PR module 2 along with the IKEY and first 16 bytes of 24 bytes of data.

6. Once the entire packet has been received from rx_ififo, and the RX parser rx_parser has written IKEY into PKT_PR_DATA, the RX packet resolution

interface (rx_pr_iface) sends IKEY and the data to the packet resolution module 2, provided that the packet resolution module 2 is ready to receive it.

7. The eDRAM interface (rx_edram_iface) of the receive module 5 sends each complete RXD_CELL to the eDRAM 4 in 17 cycle bursts.

8. Data is written to the eDRAM 4 in 32 byte bursts, giving a bandwidth of 256bits. Each of these 32 byte bursts (a total of 256 bytes) can be referenced by a single packet number called pbnum. The receive module 5 further includes a packet number manager (rx_pbnum_mgr) which maintains a queue of 3 packet numbers per FE port, or 24 for the GE port. The packet numbers are provided to the rx_max_iface when requested by the pm module of Fig. 2.

The timechart of Fig. 4 is a timechart showing the operation described above, with time flowing from left to right. The top line shows how the rx_ififo writes data to the memory RX_MEM 51. The middle line shows the transfer of data to the eDRAM. The lowest line shows how IKEY, the 16 bytes of data and control data (selected from PKT_DSC) is written to the packet resolution module. The transfer of the packet header and 24 bytes of data to the eDRAM 4 occurs upon receiving a signal revRKEY from the packet resolution module 2.

The above scheme is not altered fundamentally when the switch occurs between GE and FE. The changes are that the amount of data is larger in GE, so that more of the memory cells RXD_CELLs are used are used per packet, and that the parsing is different.

If the delay is longer than 88 cycles, only the performance is impacted. Packets stay longer in the RX. If this trend continues then eventually all PCT_DCC will be occupied.

Turning now to Fig. 5, the structure of the TX module 7 is shown, having a large degree of symmetry with the structure of he RX module 5, such that

corresponding items are given similar labels in Fig. 5. The queue/packet manager 6 of Fig. 1 is shown in Fig. 5 as a packet manager pm and a queue manager qm. The steps in the operation of this module are as follows:

1. New packets be transmitted through the interface 3 are fetched from the queue manager qm by an interface tx_qm_iface whenever at least one packet descriptor is available.

2. For each packet fetched by the interface tx_qm_iface, a packet descriptor PKT_DSC is allocated within tx_reg_file. This packet descriptor is allocated to an outgoing packet and is only de-allocated when the entire packet has been sent to the output first-in-first-out buffer interface tx_ofifo, and the packet released from the packet manager pm. The packet descriptor PKT_DSC contains information for each packet which is required for:

- Accessing data from the eDRAM 4 (the first pbnum and current pbnum).
- Sending data to tx_ofifo (ready for transmission, current offset in TXD_CELL, etc).
- Receiving packets from the packet manager pm (ready for release, etc).
- Basic buffer allocation (entry assigned, etc).

Unlike in the receive module 5, the storage of a frame within tx_mem_mgr is uniform with all data being stored in 32 byte cells called TXD_CELLs. There are 24 packet descriptors and 36 TXD cells. When the Ethernet port 1 is configured as 8 FE ports, all packet descriptors PKT_DSCs and TXD_CELLs are divided equally among all the FE ports.

3. The eDRAM interface tx_ed_iface requests data from the eDRAM 4 via individual request strobes tx_ed_req(1:0). Requests to the eDRAM 4 for a single packet are always in sequence. However, due to differences in instantaneous loads on two different banks of the eDRAM 4, data may be delivered out-of-order. Data is transferred in 17 cycle bursts.

4. Once a packet is ready for transmission, the interface tx_mac_iface informs the tx_ofifo that data is ready for the corresponding port. This is sent when the entire packet or 128 bytes (whichever is lower) has been received from eDRAM.

5. Once a packet has been received from the memory (eDRAM 4), tx_ed_iface sets a ready for release bit (rdy4rel) in the corresponding packet descriptor PKT_DSC. Release requests are then sent to the packet manager pm.

6. The tx_mac_iface interface transfers the data from tx_mem_mgr to the output interface tx_ofifo when the corresponding MAC of interface 1 is ready to receive the data. The tx_ofifo manages the actual interaction with each FE MAC, or the single GE MAC. In the FE case, storage within the output interface tx_ofifo is divided equally between all the FE ports.

Again, the above scheme is hardly varied as between the GE and FE cases. The RX and TX interfaces know which type of packets they are to handle from data received as PINS signals shown in Fig. 2.